

Description

`minindex(v, k, i, w)` returns in i and w the indices of the k minimums of v .

`maxindex(v, k, i, w)` does the same, except that it returns the indices of the k maximums.

`minindex()` may be called with $k < 0$; it is then equivalent to `maxindex()`.

`maxindex()` may be called with $k < 0$; it is then equivalent to `minindex()`.

Syntax

```
void minindex(real vector v, real scalar k, i, w)
```

```
void maxindex(real vector v, real scalar k, i, w)
```

Results are returned in i and w .

i will be a *real colvector*.

w will be a $K \times 2$ *real matrix*, $K \leq |k|$.

Remarks and examples

Remarks are presented under the following headings:

Use of functions when v has all unique values
Use of functions when v has repeated (tied) values
Summary

Remarks are cast in terms of `minindex()` but apply equally to `maxindex()`.

Use of functions when v has all unique values

Consider $v = (3, 1, 5, 7, 6)$.

1. `minindex(v, 1, i, w)` returns $i = 2$, which means that $v[2]$ is the minimum value in v .
2. `minindex(v, 2, i, w)` returns $i = (2, 1)'$, which means that $v[2]$ is the minimum value of v and that $v[1]$ is the second minimum.
- ...
5. `minindex(v, 5, i, w)` returns $i = (2, 1, 3, 5, 4)'$, which means that the ordered values in v are $v[2]$, $v[1]$, $v[3]$, $v[5]$, and $v[4]$.
6. `minindex(v, 6, i, w)`, `minindex(v, 7, i, w)`, and so on, return the same as (5), because there are only five minimums in a five-element vector.

When v has unique values, the values returned in w are irrelevant.

- In (1), w will be (1, 1).
- In (2), w will be (1, 1\2, 1).
- ...
- In (5), w will be (1, 1\2, 1\3, 1\4, 1\5, 1).

The second column of w records the number of tied values. Since the values in v are unique, the second column of w will be ones. If you have a problem where you are uncertain whether the values in v are unique, code

```
if (!allof(w[,2], 1)) {
    /* uniqueness assumption false */
}
```

Use of functions when v has repeated (tied) values

Consider $v = (3, 2, 3, 2, 3, 3)$.

1. `minindex(v, 1, i, w)` returns $i = (2, 4)'$, which means that there is one minimum value and that it is repeated in two elements of v , namely, $v[2]$ and $v[4]$.

Here, w will be (1, 2), but you can ignore that. There are two values in i corresponding to the same minimum.

When $k=1$, `rows(i)` equals the number of observations in v corresponding to the minimum, as does $w[1, 2]$.

2. `minindex(v, 2, i, w)` returns $i = (2, 4, 1, 3, 5, 6)'$ and $w = (1, 2\3, 4)$.

Begin with w . The first row of w is (1, 2), which states that the indices of the first minimums of v start at $i[1]$ and consist of two elements. Thus the indices of the first minimums are $i[1]$ and $i[2]$ (the minimums are $v[i[1]]$ and $v[i[2]]$, which of course are equal).

The second row of w is (3, 4), which states that the indices of the second minimums of v start at $i[3]$ and consist of four elements: $i[3]$, $i[4]$, $i[5]$, and $i[6]$ (which are 1, 3, 5, and 6).

In summary, `rows(w)` records the number of minimums returned. $w[m, 1]$ records where in i the m th minimum begins (it begins at $i[w[m, 1]]$). $w[m, 2]$ records the total number of tied values. Thus one could step across the minimums and the tied values by coding

```
minindex(v, k, i, w)
for (m=1; m<=rows(w); m++) {
    for (j=w[m,1]; j<w[m,1]+w[m,2]; j++) {
        /* i[j] is the index in v of an mth minimum */
    }
}
```

3. `minindex(v, 3, i, w)`, `minindex(v, 4, i, w)`, and so on, return the same as (2) because, with $v = (3, 2, 3, 2, 3, 3)$, there are only two minimums.

Summary

Consider $\text{minindex}(v, k, i, w)$. Returned will be

$$w = \begin{bmatrix} i_1 & n_1 \\ i_2 & n_2 \\ \cdot & \cdot \\ \cdot & \cdot \end{bmatrix} \quad w : K \times 2, \quad K \leq |k|$$

$$i = \left. \begin{array}{l} \begin{bmatrix} j_1 \\ j_2 \\ j_3 \\ j_4 \\ \cdot \\ \cdot \\ \cdot \end{bmatrix} \leftarrow i[i_1] \text{ is start of first minimums} \\ \begin{bmatrix} j_4 \\ \cdot \\ \cdot \\ \cdot \end{bmatrix} \leftarrow i[i_2] \text{ is start of second minimums} \\ \text{etc.} \\ i : 1 \times m, \quad m = n_1 + n_2 + \dots \end{array} \right\} \begin{array}{l} \text{has } n_1 \text{ values} \\ \text{has } n_2 \text{ values} \end{array}$$

j_1, j_2, \dots , are indices into v .

Conformability

$\text{minindex}(v, k, i, w), \text{maxindex}(v, k, i, w)$:

input:

v : $n \times 1$ or $1 \times n$
 k : 1×1

output:

i : $L \times 1, \quad L \geq K$
 w : $K \times 2, \quad K \leq |k|$

Diagnostics

$\text{minindex}(v, k, i, w)$ and $\text{maxindex}(v, k, i, w)$ abort with error if i or w is a view.

In $\text{minindex}(v, k, i, w)$ and $\text{maxindex}(v, k, i, w)$, missing values in v are ignored in obtaining minimums and maximums.

In the examples above, we have shown input vector v as a row vector. It can also be a column vector; it makes no difference.

In $\text{minindex}(v, k, i, w)$, input argument k specifies the number of minimums to be obtained. k may be zero. If k is negative, $-k$ maximums are obtained.

Similarly, in $\text{maxindex}(v, k, i, w)$, input argument k specifies the number of maximums to be obtained. k may be zero. If k is negative, $-k$ minimums are obtained.

$\text{minindex}()$ and $\text{maxindex}()$ are designed for use when k is small relative to $\text{length}(v)$; otherwise, see $\text{order}()$ in [M-5] **sort()**.

Also see

[M-5] [minmax\(\)](#) — Minimums and maximums

[M-4] [Utility](#) — Matrix utility functions

Stata, Stata Press, and Mata are registered trademarks of StataCorp LLC. Stata and Stata Press are registered trademarks with the World Intellectual Property Organization of the United Nations. StataNow and NetCourseNow are trademarks of StataCorp LLC. Other brand and product names are registered trademarks or trademarks of their respective companies. Copyright © 1985–2025 StataCorp LLC, College Station, TX, USA. All rights reserved.

For suggested citations, see the FAQ on [citing Stata documentation](#).

